2009 First International Conference on Evolving Internet

Packet Switch Architectures for Very Small Optical RAM

Onur Alparslan, Shin'ichi Arakawa, Masayuki Murata
Graduate School of Information Science and Technology
Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
$\{$*a-onur,arakawa,murata*$\}$*@ist.osaka-u.ac.jp*

***Abstract*—One of the difficulties of optical packet switched (OPS) networks is buffering optical packets in the network. The traditional rule-of-thumb buffering approach needs huge buffers due to ultra high speed of optical networks. However, optical RAM is still under research and it is not expected to have large capacity, soon. The burstiness of Internet traffic causes high packet drop rate and low utilization in small buffered OPS networks. In this paper, we investigate and compare many optical switch architectures and pacing algorithms for minimizing the buffer size of OPS switches. We show that our XCP-based pacing control with shared buffering switch architecture gives a high TCP goodput when very small optical RAM buffers are used.**

***Keywords*-small buffer; OPS; Optical RAM; TCP; switch**

I. Introduction

A well-known problem in realizing optical packet switched (OPS) networks is buffering. Recent advances in the optical networks such as dense wavelength division multiplexing (DWDM) allowed to achieve ultra high data transmission rates in optical networks. This ultra high speed of optical networks made it necessary to do some basic operations like buffering and switching in the optical domain instead of electronic domain due to high costs and limitations of electronic buffers. However, the lack of high capacity optical RAM makes it difficult to buffer enough optical packets in optical packet switched (OPS) networks. According to a rule-of-thumb [1], buffer size of a link must be $B = RTT \times BW$, where RTT is the average round trip time of flows and BW is the bandwidth of output link, in order to achieve high utilization with TCP flows. However it requires a huge buffer size in optical routers due to ultra high speed of optical links, so this buffer size is unfeasible.

Currently, the only available solution that can be used for buffering in the optical domain is using fiber delay lines (FDLs). Contended packets are switched to long fiber lines in order to be delayed. However, FDLs have important limitations. First of all, FDLs require very long fiber lines that cause signal attenuation inside the routers. There can be a limited number of FDLs in a router due to space considerations, so they can provide a small amount of buffering. Second, FDLs provide only a fixed amount of delay. FDL buffering is possible with today's technology, so many researchers consider FDL buffers to resolve contentions in optical networks. On the other hand, optical RAM is under research (e.g., Takahashi et al. [2] and NICT project [3]) and it may be available in the near future. Basic operation of optical RAM is experimentally confirmed for 40 Gbit/s 16 bit optical packets in [2]. Optical RAM solves the problems of FDLs like lack of real $O(1)$ reading operation, signal attenuation, and bulkiness. Furthermore, optical RAM is expected to have a low power consumption rate that is an important problem for electronic RAM. However, optical RAM is not expected to have a large capacity, soon. Therefore, decreasing the huge buffer requirements of OPS networks is a must in order to make use of optical buffering.

Recently, Appenzeller et al. [4] showed that when there are many TCP flows sharing the same link, a buffer sized at $B = \frac{RTT \times BW}{\sqrt{n}}$, where n is the number of TCP flows passing through the link, is enough for achieving high utilization. However, a significant decrease in buffer requirements is possible only when there are many flows on the link. This buffer requirement is still high for high-speed OPS routers with very small amount of buffering capacity. However, bursty nature of TCP flows causes a high packet drop rate in small buffered networks and limits further decreasing the buffer size. Enachescu et al. [5] proposed that $O(\log W)$ buffers are sufficient where W is the maximum congestion window size of flows when packets are sufficiently paced by replacing TCP senders with Paced TCP [6] or by using slow access links. Pacing is defined as transmitting ACK (data) packets according to a special criteria, instead of transmitting immediately upon arrival of a data (ACK) packet [6]. However, $O(\log W)$ buffer size depends on the maximum congestion window size of TCP flows that may change in time. Moreover, using slow access links is not a preferred solution when there are applications that require high-bandwidth on the network. Using Paced TCP for these applications by replacing TCP senders with paced versions can be hard. Furthermore, this proposal was based on the assumption that most of the IP traffic is from TCP flows. A recent paper [7] shows that even small quantities of bursty real-time traffic can interact with well-behaved TCP traffic and increase the buffer requirements.

It may be better to design a general architecture for OPS networks that:

- Can achieve high utilization in a small buffered OPS

network independent of the number of TCP or UDP flows;
- Does not require limiting the speed of access links;
- Does not require replacing sender or receiver agents of computers using the network.

In Ref.[8], we introduced an all-optical OPS network architecture that can achieve high utilization and low packet drop ratio by using small buffering. We evaluated the packet drop ratio in NSFNET topology using small optical RAM buffering based on our architecture in [9]. We consider an OPS domain where packets enter and exit the OPS domain through edge nodes. We proposed using a Explicit Congestion Control Protocol(XCP) based [10] intra-domain congestion control protocol for achieving high utilization and low packet drop ratio with small buffers. XCP [10] is a new congestion control algorithm using a control theory framework. XCP was specifically designed for high-bandwidth and large-delay networks. XCP was first proposed in [10] as a window-based reliable congestion and transmission control algorithm. We selected XCP framework because it allows individual control of the utilization level of each wavelength. In our architecture, when there is traffic between an edge source-destination node pair, a rate-based XCP macroflow is created, and incoming TCP and UDP packets of this edge pair are assigned the XCP macroflow similar to TeXCP [11]. The edge nodes of OPS network apply leaky-bucket pacing to the macroflows by using the rate information provided by XCP for minimizing the burstiness. As a result, there is no need to modify the TCP and UDP agents of computers or limit the speed of access links for decreasing burstiness.

Switching fabric size is an important cost factor in routers. Many switching fabric architectures like MEMS, optomechanical, electrooptic, thermooptic, liquid-crystal based switches are proposed for optical switching [12]. However, the number of switching elements in the fabric increases together with the overall cost as the number of ports of the switch increases. Also increasing the switch size introduces high crosstalk and insertion losses in many proposed switching fabric architectures. These losses require optical amplification that further increases the overall cost as explained in [12].

In our previous papers, we evaluated the performance of our proposed architecture mainly with UDP-based traffic and output buffering [8][9]. In this paper, we propose and investigate different optical switch architectures for further minimizing the size of optical switching fabric of core nodes while achieving higher goodput with small-sized optical RAM buffers. We evaluate the optical RAM requirements of our proposed architecture on a realistic mesh NSFNET topology with TCP traffic. We also compare the performance of our architecture with the method of replacing TCP with a paced version that is the generally proposed solution for small buffered networks. Simulations show that our proposed architecture with standard TCP has even better performance than Paced TCP when buffer is small.

The rest of the paper is organized as follows. Section 2 describes the basics of XCP algorithm and switch architectures. Section 3 describes the simulation methodology and presents the simulation results. Finally, we conclude and describe our future work in Section 4.

II. Architecture

In this section, we will describe the basics of XCP algorithm and switch architectures

A. XCP Basics

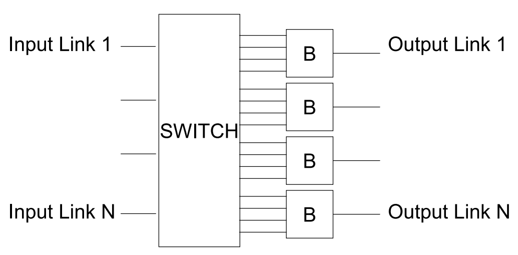
XCP is a new congestion control algorithm specifically designed for high-bandwidth and large-delay networks. XCP makes use of explicit feedbacks received from the network. Core routers are not required to maintain per-flow state information. Each XCP core router updates its control decisions calculated by an Efficiency Controller and a Fairness Controller when timeout of a per-link control-decision timer occurs.

Efficiency Controller (EC) controls the input aggregate traffic in order to maximize link utilization. A desired increase or decrease in aggregate traffic for each output port is calculated by using the equation $\Phi = \alpha \cdot S - \beta \cdot Q/d$, where Φ is the total amount of desired change in input traffic, α and β are spare bandwidth control and queue control parameters, respectively and d is the control decision interval. S is the spare bandwidth that is the difference between the link capacity and input traffic in the last control interval. Q is the persistent queue size.

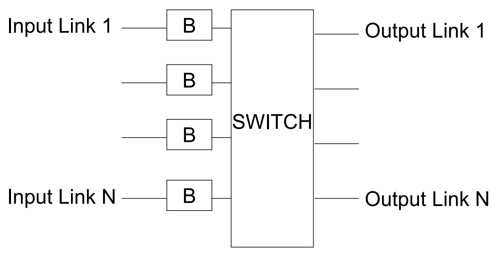
After calculating the aggregate feedback Φ, Fairness Controller (FC) fairly distributes this feedback to flows according to an AIMD-based control. However, convergence to fairness may take a long time when Φ is small. Bandwidth shuffling, which redistributes a small amount of traffic among flows, is used in order to solve this problem. Amount of shuffled traffic is calculated by $h = max(0, \gamma \cdot u - |\Phi|)$, where γ is the shuffling parameter and u is the rate of aggregate input traffic in the last control interval.

B. Switch Architectures

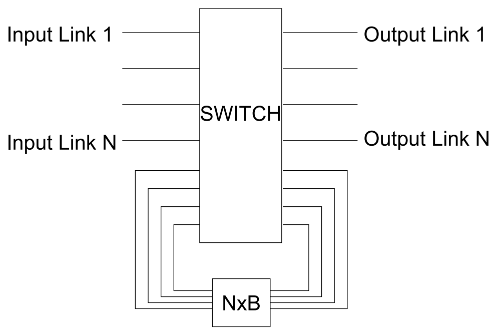
Switching fabric size is usually one of the biggest factors determining overall router cost. In this paper, we compare the performance of output buffering (OB), input buffering (IB), shared buffering (SB) and worst case shared buffering (WCSB) shown in Figure 1. Internal speedup is 1 in all switches. Output buffering has a large switch size of NxN^2 in order to prevent internal blocking where N is the nodal degree as seen in Figure 1a. It has a buffer size B at each output link. Input buffering has a switch size of only NxN as seen in Figure 1b, so it has the smallest switch size. However, a well-known problem of input buffering is head-of-line blocking, which limits the achievable utilization.



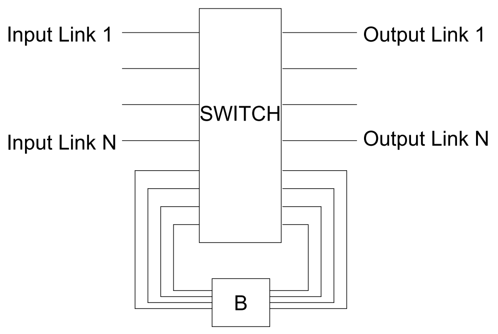
(a) Output Buffering (OB)



(b) Input Buffering (IB)



(c) Shared Buffering (SB)



(d) Worst-case Shared Buffering (WCSB)

Figure 1. Switch and FDL architectures.

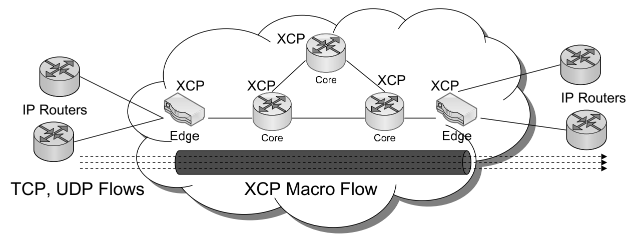


Figure 2. XCP

We apply virtual output queuing (VOQ) scheduling for minimizing this problem. Input buffering has a buffer size B at each input link. Shared buffering and worst-case shared buffering has a switch size of $2Nx2N$. Shared buffering has a single buffer with a size of $N \cdot B$, so its buffer size increases linearly with the nodal degree. Shared buffering has the same total buffer size per node as input and output buffering, so it is a more fair comparison with them at the same B value in the simulations. Only the buffer placement is different. Worst-case shared buffering has a single buffer with a size of B in dependent of nodal degree, so it has the buffer capacity of only a single link in input or output buffering.

C. Optical Rate-based Paced XCP

In [8], we proposed Optical Rate-based Paced XCP as an intra-domain traffic shaping and congestion control protocol in an OPS network domain. In this architecture, XCP sender agent on an ingress edge node multiplexes incoming flows destined to the same egress edge node and creates a macroflow as shown in Figure 2, and applies pacing with rate control to the macroflow according to XCP rate calculation.

XCP feedbacks are carried in separate probe packets that XCP sender agents send only once in every control period. There is no feedback information carried in header of data packets, so there is no need for calculating a per-packet feedback in core routers unlike in original XCP [10]. We are separating the control channel and data channels. Probe packets are carried on a separate single control wavelength that is slow enough for carrying only probe packets. Low transmission rate of control wavelength allows applying electronic conversion for updating the probe feedback and buffering the probe packets in electronic RAM in case of a contention.

When a probe packet of macroflow i arrives to a core router, the XCP agent responsible for controlling the wavelength of macroflow i calculates a positive feedback p_i and a negative feedback n_i for macroflow i. Positive feedback is calculated by

$$p_i = \frac{h + max(0, \Phi)}{N} \tag{1}$$

and negative feedback is calculated by

$$n_i = \frac{u_i \cdot (h + max(0, -\Phi))}{u} \tag{2}$$

where N is the number of macroflows on this wavelength, u_i is the traffic rate of flow i estimated and sent by the

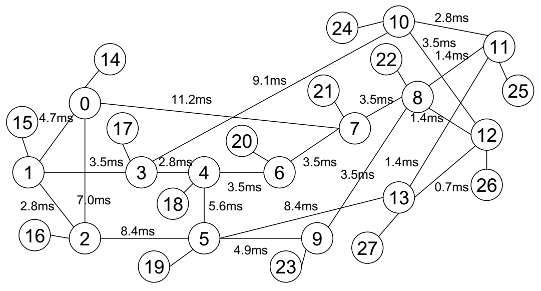


Figure 3. NSFNET topology.

XCP sender in the probe packet and h is the shuffled bandwidth. N can be estimated by counting the number of probe packets received in the last control interval. Another possible method is using the number of LSPs if GMPLS is available [11]. Control interval is the maximum RTT in the network. Control interval can be selected a bit longer than the maximum RTT for in order to compensate for processing and buffering delays of control packets. Feedback, which is the required change in the flow rate, is calculated by $feedback = p_i - n_i$. If this feedback is smaller than the one in the probe packet, core router replaces the feedback in the probe packet with its own feedback. Otherwise, core router does not change the feedback in the probe packet.

III. Evaluation

In this section we will describe the simulation methodology and present the simulation results

A. Simulation Settings

Proposed network architecture and buffering models are implemented over *ns* version 2.32 [13]. Figure 3 shows the simulated NSFNET topology. The nodes numbered from 0 to 13 are the core nodes and the rest are the edge nodes connected to the core nodes. All links (including edge and core links) have a single data wavelength and the same XCP target utilization. The propagation delay of links between core and edge nodes is selected as 0.1ms. XCP control period of core routers and probe packet sending interval of edge routers is selected as 50ms by taking extra processing and queuing delays in the core routers into account. The capacity of the data and XCP control wavelengths are set to 1Gbps and 100Mbps, respectively.

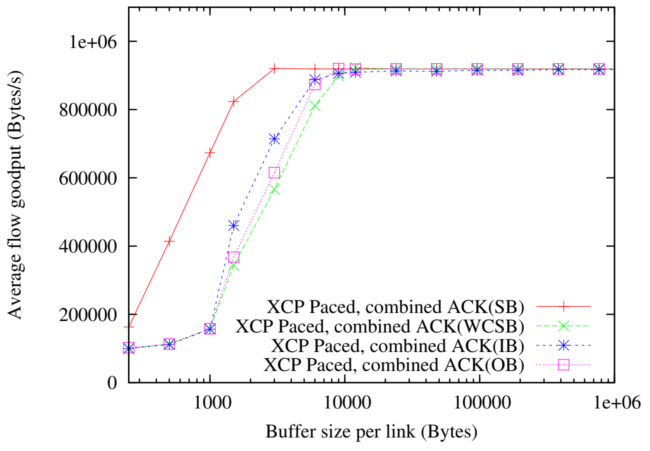
Simulator uses cut-through packet switching and buffering for data wavelengths. There is a single store-and-forward switching slow control wavelength dedicated for probe packets. Edge nodes use 0.25s of electronic buffering that is a commonly used buffer size on the Internet. However, core routers use small only optical RAM for buffering optical packets on data wavelengths. Contention of probe packets on control wavelength is resolved by electronic RAM as O/E/O conversion is not a problem for control wavelength due to its low speed.

TCP traffic is applied between edge nodes of the network. Throughout the simulation, 1587 TCP flows enter the network between randomly selected edge node pairs according to a poisson arrival process. We chose XCP's α, β and γ parameters as 0.2, 0.056, and 0.1, respectively, as explained and used in [8]. Total simulation duration is 40s. Only the simulation results in the last 5s are used for evaluation. TCP data packet size is 1500Bytes. Congestion window size limit is 20 packets. Target utilization (TU) parameter of XCP is set to 100% at both core and edge links.

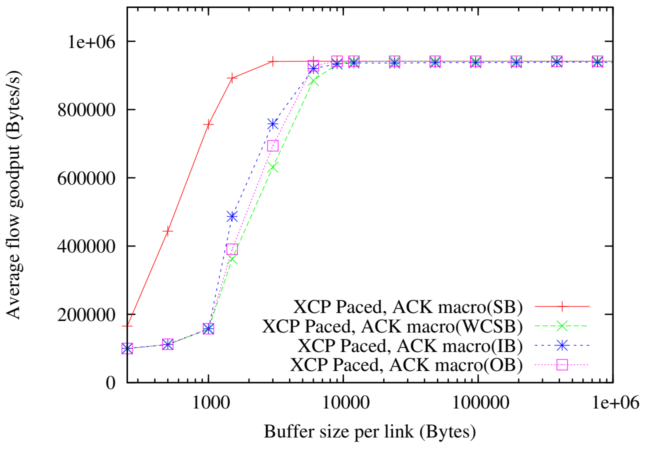
We simulate the NSFNET topology with the same standard TCP traffic and compare the performance difference with and without our XCP-based architecture. Furthermore, we do comparison with the case when TCP is replaced with its paced version that is the most common proposed method in the literature for achieving high utilization in small buffered networks.

B. Evaluations

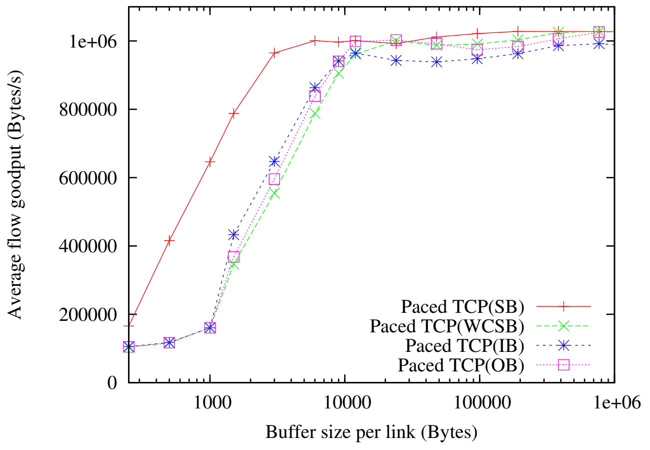
Figure 4 shows the average goodput of TCP flows on different switch and network architectures based on the optical RAM buffer size per link. In all figures, x-axis shows the buffer size per link, which is donated as B in Figure 1, in log scale and y-axis shows the average TCP goodput in linear scale. Figure 4a shows the TCP goodput when our optical rate-based paced XCP architecture is used with standard TCP Reno traffic. We see that input, output and worst case shared buffering give similar goodput. They give almost the same goodput when the buffer size is less than a single data packet or more than around six data packets. On the other hand shared buffering gives a much higher goodput than the others even though its per node buffer capacity is the same as input and output buffering. If we can use a single shared buffer instead of splitting it to input or output links, it clearly gives a much higher goodput as the small buffering capacity is used with maximum efficiency. When we compare input and output buffering, we see that when the link buffer size is between one to six data packets, input buffering gives higher goodput while using the smallest switch size among switch architectures. It is an expected result, because input buffering can handle packet connections better than output buffering when the input traffic is smooth enough. For example, let us assume that we have a switch with only single packet capacity output buffers. In case of contention of five packets coming from five input links and going to the same output link, if the buffers and links are idle, one packet will be sent to output link, one packet will be buffered in the output buffer and the rest of three packets will be dropped as there is no more buffering space. However, if we use an input switch, one packet will be sent to output link and the other four packets will be buffered at the input ports. Buffered packets can



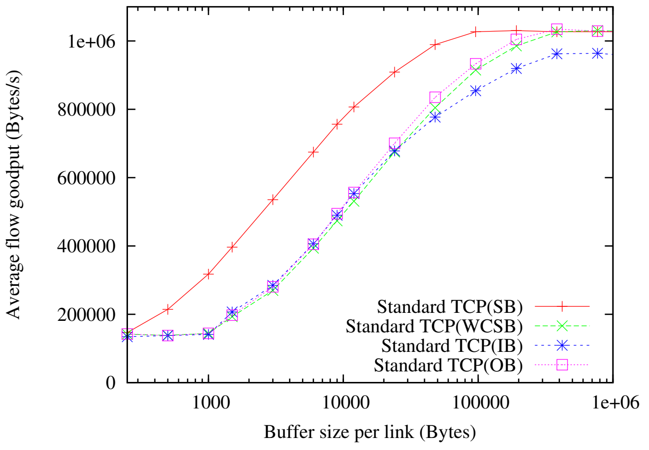
(a) XCP Pacing (combined ACK)



(b) XCP Pacing (Separate ACK Macroflow)



(c) TCP Pacing



(d) Standard TCP

Figure 4. Simulation Results

be sent to the output link as the link gets idle, so its packet dropping tendency in case of contention is lower than output buffering. Input buffering highly benefits from pacing as it decreases the back-to-back packet contentions and need of buffering at the same input link. When we check the goodput of worst-case of buffering, we see that it is very close to output buffering even though the whole switch in worst-case buffering has the same buffer capacity of only a single link in output buffering. In other words, worst case shared buffering gives almost the same goodput as output buffering with a much smaller buffer capacity per switch.

A packet level tracing of the simulation result of our proposed architecture revealed that actually there is still room for improvement. We saw that the well-known ACK compression problem [14] causes some utilization inefficiency. In our architecture it is possible to solve this problem and increase utilization by simply using separate XCP macroflows for TCP ACK packets. Figure 4b shows the TCP goodput when our optical rate-based paced XCP architecture is used with separate XCP macroflows for TCP ACK packets on the same wavelength. We see that TCP goodput gets higher than XCP with combined ACK architecture in Figure 4a. Figure 4c shows the TCP goodput when Paced TCP Reno is used without XCP control. We see that its goodput pattern is very similar to Figure 4a and 4b until buffer size per link is less than around 6 data packets. When buffer size per link is bigger than 6 packets, simulations give some fluctuating results. When the buffer is large, the achievable goodput of input buffering is lower than other switch architectures as internal blocking of the switch becomes a limiting factor at high utilization.

Figure 4d shows the TCP goodput when standard TCP Reno is used without XCP control. We see that it has much lower goodput than the simulated paced architectures. More than 10 times buffering is necessary in order to achieve high utilization equal to paced architectures. When the buffer

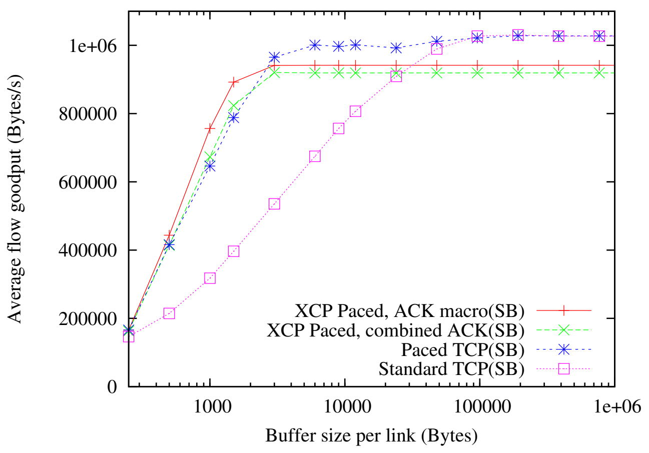


Figure 5. Overall comparison of shared buffering.

is small, goodput of input buffering is almost the same the same as output buffering which shows that pacing is necessary to pass the goodput of output buffering. Moreover, the internal blocking of the input buffered switch limits the achievable TCP goodput when the buffer is large.

Shared buffering has the highest goodput at all simulated architectures so as a next step we do a general comparison of shared buffering. Figure 5 shows the average goodput of XCP paced standard TCP, Paced TCP and standard TCP on a shared buffered switch architecture based on the optical RAM buffer size per link. We see that XCP Pacing with separate ACK macroflows gives the highest utilization when the buffer size is less than around two data packets.

IV. Conclusion and Future Work

In this paper, we showed the performance of paced and non-paced transmission control protocols on different optical switch architectures. By using a mesh topology and applying TCP traffic, simulations show that even with a traffic based on standard TCP flows, our architecture based on XCP pacing gives higher TCP goodput than the proposal of replacing TCP on the Internet with a paced version, which is the generally proposed solution in the literature. Moreover, our proposal is much easier to realize than Paced TCP as it may be difficult to replace TCP stack of computers with a paced version on the Internet in order to fully make use of small optical RAM buffered OPS networks in the future. Our proposal operates only at the edge/core routers of OPS domains and there is still no optical RAM buffered OPS network deployed on the Internet. It is enough to apply our proposed XCP architecture on OPS networks when they become commercially available.

When we compare the small buffered switch architectures, we see that shared buffering gives much higher TCP goodput than input and output buffering as it uses the buffer more effectively. However, it may be necessary to split the buffers to input or output links. In this case, if the traffic is paced, input buffering with VOQ scheduling has a higher TCP goodput than output buffering while using a much smaller switch. The drawback of VOQ scheduling of input buffering is increased scheduler complexity and processing requirements. The worst case shared buffering gives almost the same goodput as output buffering with a much smaller buffer capacity per switch. As a result, output buffering looks like the worst choice as a switch architecture for small buffered optical network on NSFNET topology. However, NSFNET core nodes mostly have a small nodal degree of 3 or 4. A topology with higher nodal degree may give different results, especially for worst-case shared buffering.

As a future work, we will simulate bigger topologies with higher nodal degree like Abilene topology and other state of the art congestion control algorithms in order to have a broader understanding on the effect of congestion control algorithms, pacing methods and switch architectures.

Acknowledgment

This work is partly supported by the National Institute of Information and Communications Technology (NICT).

References

[1] C. Villamizar and C. Song, “High performance TCP in ANSNET,” *Computer Communication Review*, vol. 24, no. 5, pp. 45–60, 1994.

[2] R. Takahashi, T. Nakahara, K. Takahata, H. Takenouchi, T. Yasui, N. Kondo, and H. Suzuki, “Photonic random access memory for 40-Gb/s 16-b burst optical packets,” *IEEE Photonics Technology Letters*, vol. 16, pp. 1185–1187, 2004.

[3] T. Aoyama, “New generation network(NWGN) beyond NGN in Japan,” Web page: http://akari-project.nict.go.jp/document/INFOCOM2007.pdf, 2007.

[4] G. Appenzeller, J. Sommers, and N. McKeown, “Sizing router buffers,” in *Proceedings of ACM SIGCOMM*, 2004, pp. 281–292.

[5] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, “Part III: Routers with very small buffers,” *ACM SIGCOMM Computer Communication Review*, vol. 35, pp. 83–90, 2005.

[6] L. Zhang, S. Shenker, and D. D. Clark, “Observations on the dynamics of a congestion control algorithm: The effects of two-way traffic,” in *Proceeedings of ACM SIGCOMM*, 1991, pp. 133–147.

[7] G. Theagarajan, S. Ravichandran, and V. Sivaraman, “An experimental study of router buffer sizing for mixed TCP and real-time traffic,” in *Proceedings of IEE ICON*, 2006.

[8] O. Alparslan, S. Arakawa, and M. Murata, “Optical rate-based paced XCP for small buffered optical packet switching networks,” in *Proceedings of PFLDnet*, February 2006, pp. 117–124.

[9] ——, "Rate-based pacing for optical packet switched networks with very small optical RAM," in *Proceedings of IEEE Broadnets*, September 2007.

[10] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product," in *Proceedings of ACM SIGCOMM*, 2002, pp. 42–49.

[11] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the tightrope: Responsive yet stable traffic engineering," in *Proceedings of ACM SIGCOMM*, 2005, pp. 253–264.

[12] G. Papadimitriou, C. Papazoglou, and A. Pomportsis, "Optical switching: Switch fabrics, techniques, and architectures," *Journal of Lightwave Technology*, vol. 21, no. 2, pp. 384–405, 2003.

[13] S. McCanne and S. Floyd, "ns Network Simulator," Web page: http://www.isi.edu/nsnam/ns/, July 2002.

[14] J. C. Mogul, "Observing TCP dynamics in real networks," in *Proceedings of ACM SIGCOMM*, 1992, pp. 305–317.